Multi-boid Flocking with Formation Control Using Local Sensing and Communication

RIMAL Biman *, *ZHEN Ziyang*, *AZEEM Muhammad*

College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, P. R. China

(Received 12 May 2020; revised 23 July 2020; accepted 30 July 2020)

Abstract: Behavior-based flocking has got remarkable attention in the recent past. The flocking algorithms can have inherent properties like organizing, healing and re-configuring for a distributed system. In this research we presented the emergent flocking behavior-based control. We defined the basis behavior and with variety of combination, and obtained a complex group behavior flocking. Unlike classical flocking, we implemented additional rules obstacle avoidance, formation and seek target which results in V-formation flocking while avoiding obstacles. We performed the visual simulation of our flocking algorithm using MATLAB. The results concluded that the multi-boid flock could successfully navigate to the target while avoiding collisions. This can be applied to areas where we need to maximize the coverage of sensors or minimize the risk of combative attack, both in military and civilian scenarios.

Key words: flocking; behavior based control; *V*-formation; motor schema; obstacle avoidance

CLC number: V279 **Document code**: A **Article ID**: 1005-1120(2020)S-0058-09

0 Introduction

In this work we deal with behavior-based flocking with formation control in swarm of multi-boid system. The goal of formation control is to maintain specific formation while boids are moving collectively in a flock. The major objective of flocking in formation control is to have multiple boids achieve a common goal, such as organizing themselves in a pre-specified geometrical shape, using their mutual information and updating their position and velocity with respect to one another and moving cohesively as one. Behavior based-control draws inspiration from biology for its design. A behavior is mapping of sensory inputs to a pattern of motor actions that are used to achieve a task[1-3]. Behaviors serve as basic building blocks for boids actions. The system is built in the bottom-up method[1]. First, basis sets of behaviors are defined. Then they are combined to obtain the desired outcome. It is supposed that a desired collective behavior emerges from the interactions between the boids and interactions of boids with the environment.

The objective of this paper is to obtain a basic understanding of basis behaviors, the design approaches related to behavior-based robotic systems, create a complex behavior by fusing basis behaviors, hence make a flocking of boids with desired formation and study its characteristics.

Flocking in formation is inspired from biology, the collective motions of animals in groups. This synchronized and collective movement assists animals to stay together when searching for food, defending against predator, and staying together while migration. Nature flocks consist of two balanced, opposing behaviors: Staying close to a flock while avoiding collisions within the flock[4-5]. A flocking holds some properties. It is made up of discrete agents; its overall motion seems fluid; it is simple in concept but visually complex; it is randomly ar-

*Corresponding author, E-mail address: biman_rimal@nuaa.edu.cn.

How to cite this article: RIMAL Biman, ZHEN Ziyang, AZEEM Muhammad. Multi-boid flocking with formation control using local sensing and communication[J]. Transactions of Nanjing University of Aeronautics and Astronautics, 2020, 37(S): 58-66.

http://dx.doi.org/10.16356/j.1005-1120.2020.S.008

ranged but highly synchronized; it seems intentional with centralized control but it is only due to aggregate result of individual boids. Possible formations can be "hard-coded", in the sense that they specify cartesian positions for all boids[3], or formations can be defined by constraints, which allows variation in cartesian positions for robots like maintaining line-of-sight and visibility. Basic needs for flocking in formation are each boid's ability to sense nearby boid's position, direction and velocity. It also needs the ability to sense nearby obstacles and distinguish between boids and obstacles. These are obtained from sensors like visual recognition, beacons, global positioning system (GPS) and radio broadcast.

The problem addressed in this paper is useful to military application; especially, the proposed target is useful for drone swarm platoon or scout unit. Further, it can be applied to areas where we need to maximize cover of sensors or minimize the risk of adversarial attack. Nowadays, many military and noncombatant applications involve a group of agents to cooperate to complete a specific mission autonomously. Examples of civil uses include forest fire monitoring and fighting[6], buildings and bridges inspection, crop dusting and search and rescue of survivors after a disaster, and military uses include surveillance and monitoring of an area.

The control design of flocking information systems is quite complex because a lot of issues are needed to be considered, such as formation keeping, communication, and coordination between boids. In addition, when the number of the boids reaches a certain degree (hundreds or even more), the computational cost of the system will increase.

1 Background and Related Work

One of the pioneers to start exploring the potential and possibilities of flocking was. Reynolds[4]. Reynolds defined fundamental aspects for flocking with three empirical rules: Avoid crowding (separation), match heading (alignment), and avoid separation (cohesion). The behaviors that make up Reynolds flocking model are based on the principle that each agent relates to its neighboring agents[5]. Luo et al. also developed a flocking algorithm with multi-target tracking for multi-agent systems, which can be used for swarm preying[7].

Even very simple networks can make a robot display some forms of basic intelligent behaviors. This was noted by Braitenberg[8]. He described, in an informal way, how intelligent artificial creatures could be constructed incrementally starting from very simple ones. Direct sensor-actuator mapping can make robots display basic intelligent behaviors.

Mataric used local control laws to generate desired global behaviors[1]. The fundamental principle is to define basis behaviors as general building blocks for synthesizing the group behavior. A basis behavior is those that deal with its survival. Some of the behaviors are safe-wandering, following, aggregation, dispersion and homing.

Another work is on Schema theory. That has been used by psychologists since early 1900s . It was brought to attention of roboticists in 1980s[9]. It provides way to cast biological insights into a formalism , way of expressing basic unit of activity. Its schema consists of information on how to act and/or perceive (knowledge, data structures, models), computational process by which it achieves the activity (algorithm). It is a generic template for how to do some activity.

Formation-keeping was first brought by Balch et al. in the behavior-based control[3]. They used motor schemas and was fully integrated obstacle avoidance. They used motor schemas for formation-keeping, move-to-goal, avoid-static-obstacle, avoid-robot, maintain-formation. Balch's different formations include line, wedge, diamond and column.

McLurkin[10] developed libraries containing 40 swarm behaviors: Avoid many robots, disperse from source, disperse from leaves, disperse uniformly, compute average bearing, follow the leader, and cluster into groups. The approach to generating behaviors is similar to Mataric's, and the primary differences is that algorithms are more tuned.

The computation power of computer has been boosted; the research of this realm has obviously

evolved in recent years. Peter Stone, one member of the champion team in Robo-Cup simulation league, designed layered learning in multi-agent systems for formation planning and task assignment[1].

V-like formations in flocks of artificial birds is considered by Nathan et al.[11]. They study the emergence of V-like formations during flight and demonstrate it by means of simulations.

Ref.[12] a behavior-based control design approach is proposed for two kinds of important formation control problems: Efficient initial formation and formation control while avoiding obstacles. The swarm robots can form or keep various formations, for example, line, orthogon, and triangle[3]. In addition, the formation can transform in order to suit different tasks. There are three main methods to implement formation control, namely, behavior- based formation control[13-14], leader-follower[15-16], and virtual structure methods[17]. The behavior-based formation control method utilizes a set of predefined basic behaviors, such as moving-to-target, keeping-formation, and avoiding-obstacles.

Today's generation is of autonomous agents. Largenumber of autonomous robot algorithms are available; rapid advancement emerges in individual agents, advancement helps to understand complex systems, and most importantly, computational capabilities has increased. Size and price of sensors and chipshas has reduced and are easily available.

This work aims to achieve flocking, formation, obstacle avoidance, and to navigate to goal simultaneously. The approach has the virtue to be simple, intuitive and effective. A visual simulation in MATLAB was developed to explore the benefits and pitfalls. It can be shown working under various conditions. In this work agents are autonomous mobile boids. Autonomous agents can move around freely in unstructured environments, operating without continuous human guidance.

These agents only have minimum sensor range and communication radius with other agents. Interactions between agents lead to the emergence of global intelligent behavior. In the beginning of simulation, each boids location is generated randomly, then in a few iterations, the V-pattern is formed. They collectively move in formation towards the target goal while avoiding observed obstacles. Mostly researcher use flocking or formation only but, in this work, simultaneous flocking with V-formation is used.

2 Behavior Based Control System

This is a control strategy based on a network of behaviors that achieves and/or maintains goals. Behaviors are implemented as control laws, either in software or hardware[1]. Each behavior can take inputs from the sensors, and send outputs to the actuators.

Since classical control assumes internal representation of real world, lots of state variables, elaborate algorithms for decision making, requires a lot of memory and can be slow in reacting to sensor input and requires outside intelligence, planning, and foresight. With introduction of behavior-based robotics, Professor Rodney Brooks claimed "Planning is just a way of figuring out what to do next"[18]. In behavior based control system (BBS) we decompose the problem into many "simple" behaviors. Many simple behaviors can work together to appear complex.

In BBS control, a boid should not care for parts of the environment that are far away, unless it really needs them. Nearby boid can help by providing infomation on the environment. The local planner gathers information from all the robots in it and makes plans for them while they are nearby. The boid can ask a local planner for a plan to follow the boids who have the navigator, pilot and controller. A global planner coordinates the missions of the boids.

BBS system can be designed by two methods.

(1) Subsumption architecture

Task achieving behaviors in the subsumption architecture are represented as separate layers operating in parallel. Key principle of subsumption architecture is touse the world as its own best model. (Fig.1)

Designing in subsumption: Qualitatively specifies the overall behavior needed for the task and de-

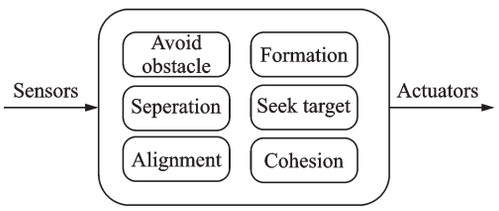

Fig.1 Information flow in behavior-based system

compose that into specific and independent behaviors (layers). The layers should be bottomup and consist of disjoint actions. Ground low-level behaviors in the robots are sensors and effectors.

This method holds some of the advantages. It reacts immediately; it is fast and have parallelism. It also have some disadvantages. The priorities must be evaluated before and require behaviors to fit into a simple single-inheritance hierarchy. Behaviors cannot be combined, only enhanced linearly. Genghis is one of the famous robots with this architecture.

(2) Motor schemas

Schemas are functional units for analysis of cooperative competition in the brain[8]. Schemas program units especially suit for a system which has continuing perception and interaction with its environment. It is a programming language for new systems in computer vision, robotics and expert systems. It is a bridging language between distributed artificial intelligence and neural networks for specific subsystems. It is based on the action-perception cycle. To put it more concisely, a schema is a behavior (Fig.2).

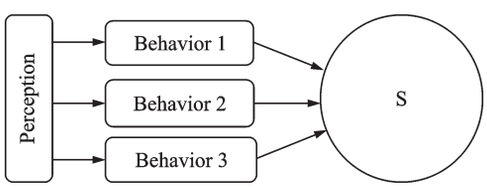

Fig.2 Behaviors via vector summation

Schema theory explains behavior in terms of interaction of many concurrent activities: Cooperative computation; a kind of computation based on the competition and cooperation of concurrently active agents; cooperation that yields a pattern of strengthened alliances between mutually consistent schema instances. Since competition cases do not meet the evolving consensus lose activity, this is not part of this solution.

The schema-based methodology has many advantages, including the use of distributed processing, which facilitates real time performance and modular construction of schemas for ease in the development, testing, and debugging of new behavioral and navigational patterns. Complex behavioral patterns can be emulated by the concurrent execution of individual rule. Some of the defined motor schemas are move ahead, move to the goal, avoid static obstacles, dodge, escape, stay-on-path, noise, follow-the-leader.

Behavior weighting: By allowing the force produced by a perceived environmental object to vary in relationship to the certainty of the object's identity (whether it is an obstacle, goal path, or else), dynamic re-planning is trivialized. Since the sensed environment produces the forces influencing the trajectory of the boid, when the perception of the environment changes, so do the forces acting on the boids, and consequently so does the boid's path. All this is accomplished at a level beneath the prior knowledge representations. Behaviors fusion is done via vector summation. No predefined hierarchy exists for coordination, and instead behaviors are configured at run-time. Pure arbitration is not used, each behavior can contribute in varying degrees to robot's overall response. Using trial and error each rule is given weight. Rule that has higher importance has a higher value. Some of the robot vehicles are Callisto a foraging robot, GT Hummer, Harv, George.

Emergent behavior: Emergent behavior is an important phenomenon found in behavior-based robotics. Robot behaviors emerge from interactions of rules, interactions of behaviors and interactions with environment. We all know sum is greater than the parts, thus emergent behavior is more than the controller that produces it.

Motor schema design systems can be designed to take advantages of emergent behavior. Using basis behaviors in a variety of combinations to enable creation of more complex group behaviors, such as flocking, foraging, surrounding, herding. Here we work on the emergent flocking system.

3 Algorithm Implementations

In each iteration, each boid obtains its position, neighborhood position through communication and force from fusion of behavioral rule to move towards the target. Forces are combined through a weighted sum to generate the final resulting direction the boids should move. Then how much that rule influences the final result determines weight applied to each rule. With change in weights can result in different behavior.

Algorithm 1 Main loop of the swarming algorithm

```
1  func SWARM
2  getPose( )
3  getNeighbors( )
4  getDesiredSpacing( )
5  gettheta ( )
6  getObstacles( )
7  getTarget( )
8  getObstacleRange( )
9  getMaxSeeAhead( )
10  f1 ← separation( )
11  f2 ← alignment( )
12  f3 ← cohesion( )
13  f4 ← obstacle_avoidance( )
14  f5 ←seekForce( )
15  f6←fromationForce( )
16  F_total ← G1 * f1 + G2 * f2 + G3 * f3 + G4 * f4 + G5 * f5 + G6 * f6
17  if F_total > F_max
18  F_total = F_max
19  Move(F_total)
20  END func
```

(1) Separation rule

This rule urges boids to move away from other boids to avoid collisions. This is done by creating a vector for each one of the detected neighbors pointing to the exactly opposite direction of that neighbor, and then adjusting this vector's magnitude to be bigger if the neighbor is closer. The vectors are summed into one resultant vector that is returned to the main program.

Algorithm 2 Separation rule

```
1  func SEPARATION:
2  Vector f ← 0
3  for all neighbors n do
4  f_n ← this.position − n.position
5  f_n.normalize( )
6  f_n ← f_n/distance(this, n)
7  f ← f − n
8  end for
9  return v
10  end func
```

(2) Cohesion rule

This rule tries to move the boids to the center of mass of the neighbor boids, to form a swarm. We calculate the average position of the neighbors. This is the group's center of mass. We create a vector pointing to this location and return it to the main program.

Algorithm 3 Cohesion rule

```
1  func COHESION
2  Vector f ← 0
3  for all neighbors n do
4  f ← f + n.position
5  end for
6  f ← f/neighbors.length
7  return f
8  end func
```

(3) Alignment rule

This rule urges the boid to move in the same direction as its neighbors are moving. We loop through each detected neighbor and calculate the average direction they are moving using their heading. We then return this vector to the main function.

Algorithm 4 Alignment rule

```
9  func ALIGNMENT
10  Vector f ← 0
11  for all neighbors n do
12  f ← f + n.velocity
13  end for
14  f ← f/neighbors.length
15  return f
16  end func
```

(4) Obstacle avoidance rule

This rule urges boids to move around the obstacle. We provide boid's sensor to see ahead and when boid detect an obstacle in the path then we apply appropriate steering forces to avoid obstacles in

their paths. We will consider a simple idealization of an obstacle: a circle.

Algorithm 5 Obstacle avoidance rule

```
1  funcobstacle_avoidance
2  for all obstacles n do
3  aheadPos( )
4  intersection_line( )
5  obstacle_check( )
6  f=ahead.position-obstacle_check
7  end for
8  Return f
9  End func
```

(5) Seek rule

This rule urges the boids to seek a target and move in the direction of the target. When boids reaches certain distance near to the target, its velocity reduces to prevent oscillations.

Algorithm 6 Seek rule

```
1  func seek
2  f= Target-this.Position
3  Slowing_radius = (Target-50)
4  If |f| < slowing_radius
5  f= (f*|f|)/(10*slowing_radius)
6  return f
7  close func
```

(6) Formation rule

This rule urges the robot to move in a V-formation when moving collectively towards the target. In order to keep a desired formation while moving, each robot should maintain a certain angle and distance to other boids, as shown in Fig.3.

Algorithm 7 Formation rule

```
1  func formation
2  for all neighbors n do
3  get_neighbor_ID
4  get_leader_id_&_position
5  if id is even
6  position_angle_60^0
7  distance_succesive_leader_50
8  If id is odd
9  position_angle_−60^0
10  distance_succesive_leader_50
11  f = position
12  return f
13  end func
```

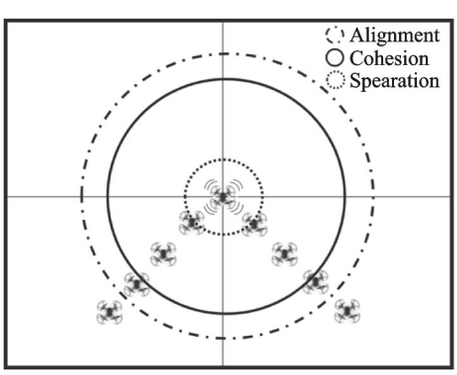

Fig.3 Boids sensor range

4 Simulation

A new artificial intelligence behavior-based robotics, made up by bottom up method is simulated in MATLAB. In this simulation each boid is represented as autonomous quadcopter. These boids only have minimum sensors with limited sensing range to communicate with other boids. There is no centralized control. Interactions between agents is local and lead to the emergence of global intelligent behavior.

In this 1 200×1 200 screen (representing the field of 1 200 m×1 200 m) obstacles are represented as circle and the target as a "*". Obstacles are created with gaussian random distribution randomly around the screen. At first the boids are generated randomly, after few iterations they start to get into the V-pattern formation. They collectively move in this formation towards the target goal. When they perceive obstacle ahead, they move around it. After crossing obstacles, they reform into V-formation. Most studies only consider flocking or formation. In this work we have simultaneous used formation as one of the rules to obtain a complex flocking in V-formation behavior. At last, after reaching the target, they form a circle around the target. Table 1 lists the parameters of the flocking boids.

In this simulation we implement the following rules: The original flocking behaviors (separation, alignment, and cohesion) and additional behaviors (goal-seeking, obstacle avoidance and formation). Various combinations of the weight factor for each rule are used in the simulations. The order of force

Table 1 Flocking boid parameters

Rule	Parameter	Value
Separation	Gain	1.5
	Sphere of influence	50
Cohesion	Gain	1.2
	Sphere of influence	300
Alignment	Gain	1
	Sphere of influence	300
Seek target	Gain	1
	Slowing radius	50
Obstacle Avoidance	Gain	1.5
	Radius of influence	100
Formation	Gain	2
	Desired spacing	50
Noise	Desired angle	60
	Wander angle	5
Limitation	Max force	1
	Max speed	3

weight factors is separation, formation, obstacle avoidance, cohesion and target seeking in descending order. Separation is given the most weight, since collisions should be avoided at all costs, generally at the expense of staying together or heading in the same direction. The simulation results are shown in Fig.4.

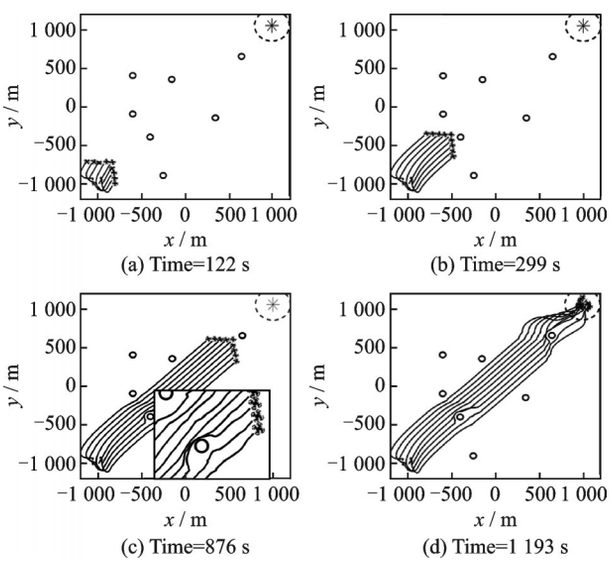

Fig.4 Flock simulation preview

The flocking size used is 9 for the simulation. Small black quadcopter generated randomly around the position(−1 000, −1 000). Every boid is given a number as its boid's ID. The ID is being updated on runtime according to the position relative to the target. They start to flock in v-formation towards the desired heading (target). When they reach around the position (−500, −500), they encounter obstacle. They avoid the obstacle by moving around left or right of it. After avoiding the obstacle, most of the agents stay in flock, while few may split away.

Finally, flocks converge in around 1 200 iterations. In each iteration velocity and position of the agent is updated according to aggregated force felt by boids. The boids move by repeating the execution cycle by getting weighted sum of force and adding it to current position. Here in this simulation, we limit the speed. Without such limitations, their speed will actually fluctuate with a flocking-like tendency, and it is possible for them to momentarily go very fast.

Visual analysis of trial runs shows promising results. In all test cases, the flock is able to navigate in such a way as to avoid collisions between the obstacles and with the other boids. Computationally, this methodology is very simple to implement, and requires very little in the way of intra boid communication with limited sensor range. These computational requirements are well within the capability of the embedded microcontrollers. At first tuning of rule weightage is required using trial and error. With subsequent simulation required weightage is given. In between simulation, boids sometime feel null force because of interaction with all the rules. So we add some noise so that it can come out from such situations. Each mobile boid is programmed with the same simple algorithm. The results show that it is possible to create complex behaviors using relatively simple algorithms.

5 Conclusions

The proposed emergent flocking behavior is simulated with all the agents in swarm moving from initial state to final state in presence of obstacles. From simple individual behavior a complex emergent behavior is achieved. The effectiveness of this approach is tested using MATLAB. The results show that the swarm can avoid the obstacle while maintaining the formation. The time it takes to ar-

rive at the target position increases as the number of swarm members increases. The reason of this is because the boids depend on neighbor's speed and direction. They wait for its members to escape from the obstacle before going to its target position. The algorithm has the advantage of being decentralized, meaning that every boid needs only information about its closest surroundings and there is no central coordinator. Future direction from this research is to implement self-learning algorithm using reinforcement learning, where boids will learn them self to reach the target in flock.

References

[1] MATARIC M J. Designing and understanding adaptive group behavior[J]. Journal Adaptive Behavior, 1995, 4(1): 51-81.

[2] TU K, CHIEN M C. Formation control of the multi-robot team's behaviors based on decentralized strategies[C]//Proceedings of the American Control Conference 2006. Minneapolis, USA:[s.n.], 2006:1-11.

[3] BALCH T, ARKIN R C. Behavior-based formation control for multirobot teams[J]. IEEE Transactions on Robotics and Automation, 1998, 14(6): 926-939.

[4] REYNOLDS C W. Flocks, herds and schools: A distributed behavioral model[C]//Proceedings of ACM SIGGRAPH Computer Graphics. [S. l.]: ACM, 1987: 25-34.

[5] SAJWAN M, GOSAIN D, SURANI S. Flocking behavior simulation: Explanation and enhancements in boid algorithm[J]. International Journal of Computer Science and Information Technologies, 2014, 5(6): 5539-5544.

[6] BRAGA R F, SILVA R C, RAMOS A C B. UAV swarm control strategies: A case study for leak detection[C]//Proceedings of 2017 the 18th International Conference on Advanced Robotics. Hongkong, China: IEEE, 2017: 173-178.

[7] LUO Xiaoyan, LI Shaobo, GUAN Xinping. Flocking algorithm with multi-target tracking for multi-agent systems[J].Pattern Recognition Letter, 2010, 31(9): 800-805.

[8] BRAITENBERG V. Vehicles: Experiments in synthetic psychology[M]. Boston, USA: The MIT Press, 1984.

[9] SCHEMA A A. Encyclopedia of artificial intelligence[M]. 2nd ed. New York, USA: John Wiley, 1992: 1427-1443.

[10] MCLURKIN J D. Stupid robot tricks: A behavior-based distributed algorithm library for programming swarms of robots[D]. Boston, USA: Massachusetts Institute of Technology, 2004.

[11] NATHAN A, BARBOSA V. V-like formations in flocks of artificial birds[J]. Artificial Life, 2014, 14(2): 179-188.

[12] XU D, ZHANG X, ZHU Z, et al. Behavior-based formation control of swarm robots[J]. Mathematical Problems in Engineering, 2014, 2014: 1214-1225.

[13] HE L L, LOU X C. Study on the formation control methods for multi-agent based on geometric characteristics[J]. Advanced Materials Research, 2013, 765/766/767: 1928-1931.

[14] CHETTY R M K, SINGAPERUMAL M, NAGARAJAN T. Behavior-based multi robot formations with active obstacle avoidance based on switching control strategy[J]. Advanced Materials Research, 2012, 433/434/435/436/437/438/439/440: 6630-6635.

[15] CHEN J, SUN D, YANG J, et al. Leader-follower formation control of multiple non-holonomic mobile robots incor- porating a receding-horizon scheme[J]. International Journal of Robotics Research, 2010, 29(6): 727-747.

[16] GHOMMAM J, MEHRJERDI H, SAAD M. Leader-follower based formation control of nonholonomic robots using the virtual vehicle approach[C]//Proceedings of the 2011 IEEE International Conference on Mechatronics. Istanbul, Turkey: IEEE, 2011: 516-521.

[17] BENZERROUK A, ADOUANE L, LEQUIEVRE L, et al. Navigation of multi-robot formation in unstructured environment using dynamical virtual structures[C]//Proceedings of the 2010 International Conference on Intelligent Robots and Systems. Taipei, China: IEEE, 2010: 5589-5594.

[18] ROOKS R A. Cambrian intelligence[M]. Boston, USA: The MIT Press,1999.

Acknowledgement The author is very grateful to College of Automation for supporting and providing laboratory for analysis.

Author Mr. **RIMAL Biman** received his B.S. engineering in Aeronautical engineering from Nanjing University of Aeronautics and Astronautics (NUAA), in 2015. He completed his Master of Engineering in Navigation, Guidance and Control from NUAA, Nanjing, China, in 2020. His research is focused on swarm control of unmanned aerial vehicle for target tracking. He engaged in designing of blended wing body of unmanned aerial vehicle during his bachelor's studies. And during masters his research focused on swarm system,

developed algorithms of swarm system in MATLAB to understand the properties of swarms.

Author contributions Prof. **ZHEN Ziyang** designed the study, revised and modified the manuscript. Mr. **RIMAL** compiled the models, conducted the analysis, interpreted the results and wrote the manuscript. Mr. **AZEEM** contributed to discussion and background of the study. All authors commented on the manuscript draft and approved the submission.

Competing interests The authors declare no competing interests.

(Production Editor: ZHANG Bei)

基于局部传感和通信的多波集群编队控制

RIMAL Biman, 甄子洋, AZEEM Muhammad
(南京航空航天大学自动化学院,南京 211106,中国)

摘要:基于行为的集群成为当前的研究热点。集群算法具有自组织、自修复、可重构等固有属性,因此可以用于分布式系统。本文介绍了基于行为法的集群涌现控制。首先定义了几种基本行为以及不同的组合形式,建立了复杂的集群聚集行为模型。与经典集群控制不同,本文引入了避障规则,使得集群个体在避开障碍物的同时,形成"V"形编队并搜寻目标。最后使用MATLAB软件对集群算法进行可视化仿真。仿真结果验证了多波集群算法能够使集群个体避开障碍物,搜索到目标所在位置。本文提出的算法可以应用于军事和民用场景中,使集群能够最大化传感器覆盖范围或减小在战场区域中被攻击的风险。

关键词:集群;基于行为的控制;"V"形编队;运动算法;避障